Introduction to Machine Learning

Regularization Introduction









Learning goals

- Overfitting
- Motivation of regularization
- First overview of techniques
- Pattern of regularized ERM formula

WHAT IS REGULARIZATION?

Methods that add **inductive bias** to model, usually some "low complexity" priors (shrinkage and sparsity) to reduce overfitting and get better bias-variance tradeoff



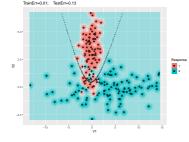
- Explicit regularization: penalize explicit measure of model complexity in ERM (e.g., L1/L2)
- Implicit regularization: early stopping, data augmentation, parameter sharing, dropout or ensembling
- Structured regularization: structural prior knowledge over groups of parameters or subnetworks (e.g., group lasso

RECAP: OVERFITTING

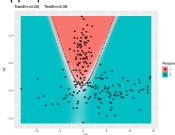
- Occurs when model reflects noise or artifacts in training data
- Model often then does not generalize well (small train error, high test error) – or at least works better on train than on test data



Overfitted model

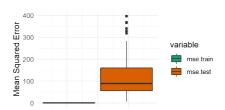


Appropriate model



EXAMPLE I: OVERFITTING

- Data set: daily maximum **ozone level** in LA; n = 50
- 12 features: time (weekday, month); weather (temperature at stations, humidity, wind speed); pressure gradient
- Orig. data was subsetted, so it feels "high-dim." now (low n in relation to p)
- LM with all features (L2 loss)
- MSE evaluation under 10 × 10 REP-CV



Model fits train data well, but generalizes poorly.



EXAMPLE II: OVERFITTING

- We train an MLP and a CART on the mtcars data
- Both models are not regularized
- And configured to make overfitting more likely

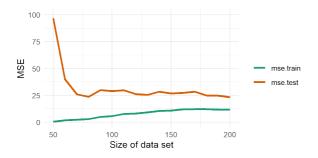
	Train MSE	Test MSE
Neural Network	3.68	19.98
CART	0.00	10.21

(And we now switch back to the Ozone example...)



AVOIDING OVERFITTING – COLLECT MORE DATA

We explore our results for increased dataset size.





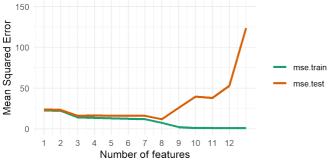
Fit slightly worsens, but test error decreases.

But: Often not feasible in practice.

AVOIDING OVERFITTING – REDUCE COMPLEXITY

We try the simplest model: a constant. So for L2 loss the mean of $y^{(i)}$.

We then increase complexity by adding one feature at a time.



NB: We added features in a specific (clever) order, so we cheated a bit.



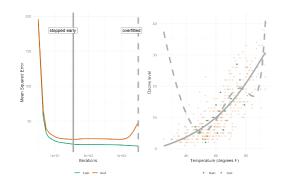
AVOIDING OVERFITTING – OPTIMIZE LESS

Now: polynomial regression with temperature as single feature

$$f(\mathbf{x} \mid \boldsymbol{\theta}) = \sum_{k=0}^{d} \theta_k \cdot (x_T)^k$$

We set d=15 to overfit to small data. To investigate early stopping, we don't analytically solve the OLS problem, but run GD stepwise.





We see: Early stopping GD can improve results.

NB: GD for poly-regr usually needs many iters before it starts to overfit, so we used a very small training set.

REGULARIZED EMPIRICAL RISK MINIMIZATION

We have contradictory goals:

- maximizing fit (minimizing the train loss)
- minimizing complexity of the model



× × ×

We saw how we can include features in a binary fashion. But we would rather control complexity **on a continuum**.

REGULARIZED EMPIRICAL RISK MINIMIZATION

Common pattern:

$$\mathcal{R}_{\text{reg}}(f) = \mathcal{R}_{\text{emp}}(f) + \lambda \cdot J(f) = \sum_{i=1}^{n} L\left(y^{(i)}, f\left(\mathbf{x}^{(i)}\right)\right) + \lambda \cdot J(f)$$



- J(f): complexity penalty, roughness penalty or regularizer
- $\lambda \ge 0$: complexity control parameter
- The higher λ , the more we penalize complexity
- $\lambda=0$: We just do simple ERM; $\lambda\to\infty$: we don't care about loss, models become as "simple" as possible
- ullet λ is hard to set manually and is usually selected via CV
- As for \mathcal{R}_{emp} , \mathcal{R}_{reg} and J are often defined in terms of θ :

$$\mathcal{R}_{\mathsf{reg}}(oldsymbol{ heta}) = \mathcal{R}_{\mathsf{emp}}(oldsymbol{ heta}) + \lambda \cdot J(oldsymbol{ heta})$$